Resource-Efficient Knowledge Distillation for Qwen2.5-0.5B-Instruct: A Stable Pipeline Using a 4-bit Quantized 3B Teacher and QLoRA

Zhaokun Wang
Institute of Computational Linguistics
Heidelberg University, Heidelberg, Germany
zhaokun.wang@stud.uni-heidelberg.de

Abstract

Deploying large language models (LLMs) is computationally expensive, making model compression techniques like knowledge distillation (KD) essential. This work presents a stable, resource-efficient KD pipeline to compress a **Qwen2.5-3B-Instruct** teacher into a **Qwen2.5-0.5B-Instruct** student on consumer-grade hardware. Our memory-conscious approach uses a 4-bit quantized teacher and trains the student with 4-bit QLoRA. The core of our method is a **token-averaged Kullback-Leibler (KL) divergence loss** with gradient clipping, which stabilizes training. On a held-out validation set, the distilled student's perplexity improved by **25.8%** (from 5.504 to 4.085), even slightly surpassing its 3B teacher (PPL 4.225). Our pipeline demonstrates an effective path for creating powerful, lightweight models under significant resource constraints. [1]

1 Introduction

While large language models (LLMs) have demonstrated remarkable capabilities across various tasks (Brown et al., 2020; Touvron et al., 2023), their substantial size imposes significant challenges for training, inference, and deployment, particularly in resource-constrained environments. Model compression techniques, such as pruning (LeCun et al., 1990), quantization (Gholami et al., 2022), and knowledge distillation (KD) (Hinton et al., 2015), are essential for making these models more accessible.

This project focuses on knowledge distillation, a process where a smaller "student" model learns to mimic the behavior of a larger "teacher" model. We distilled a 3B-parameter teacher into a 0.5B-parameter student, where both models are quantized to 4-bit precision to minimize memory footprint. This setup is designed to be executable on a single consumer-grade GPU (e.g., NVIDIA T4 or RTX series).

Our primary discovery is the refinement and validation of a stable KD pipeline. We demonstrate that transitioning from a conventional sum-based KL divergence loss to a **token-averaged KL loss**, combined with **gradient clipping** (Pascanu et al., 2013), not only stabilizes training but also yields superior performance. This revised methodology results in a student model that significantly improves upon its baseline and even slightly outperforms its larger teacher on in-distribution validation data, verifying a path for effective, low-resource LLM compression.

2 Related Work

Our work builds upon established concepts in model compression, particularly knowledge distillation and quantization.

Knowledge Distillation. Knowledge distillation (KD) is a model compression paradigm where a compact "student" model is trained to mimic the behavior of a larger, more powerful "teacher" model (Hinton et al., 2015). The goal is to transfer the generalized "dark knowledge" from the teacher, which is encoded in its output probability distributions, rather than just learning from hard labels in the data. This allows the student to achieve significantly better performance than if it were trained on the same data from scratch.

White-box vs. Black-box Distillation. KD approaches can be broadly categorized based on the accessibility of the teacher model. In white-box distillation, the student has full access to the teacher's internal components, including its parameters, hidden states, and, most commonly, the pre-softmax logits. This allows for direct alignment of the student's and teacher's output distributions using metrics like KL divergence, as proposed by Hinton et al.

[1] Codes are publicly available at https://github.com/BufferHund/ResourceEfficient_Distillation_SemesterProject

(2015). Other white-box methods, such as TinyBERT (Jiao et al., 2020), go further by matching intermediate hidden states and attention matrices to provide a richer training signal. Our work falls into this category, as we directly utilize the teacher's logits.

In contrast, black-box distillation is employed when the teacher model is only accessible via an API, exposing only its final predictions (e.g., the generated text) and not the underlying probability distributions. In this setting, a common approach is to create a synthetic dataset by querying the teacher with a set of prompts and then using the teacher's high-quality outputs as target labels to fine-tune the student model (Wang et al., 2022). This is essentially supervised fine-tuning on the teacher's generated data, without access to the soft targets that characterize white-box KD.

Quantization and Efficient Fine-Tuning. Alongside KD, quantization is another critical technique for reducing the memory footprint and computational cost of LLMs (Gholami et al., 2022). It involves representing model weights and activations with lower-precision data types, such as 4-bit integers. Our work leverages 4-bit quantization for both the teacher and student models. Specifically, we use QLoRA (Dettmers et al., 2023), a parameter-efficient fine-tuning method that allows for training a quantized model by attaching small, trainable LoRA adapters (Hu et al., 2021). This combination of KD and QLoRA forms the foundation of our resource-efficient pipeline.

3 Methodology

3.1 Dataset and Preprocessing

We utilize the `yahma/alpaca-cleaned` dataset, which is derived from the self-instruct method (Wang et al., 2022). For our primary experiments, we train on the first 5% of the dataset (`train[:5%]`) and validate on the subsequent 1% (`train[5%:6%]`). Instruction-following examples are formatted by concatenating the `instruction` and `input` fields to form the prompt. The model is trained to predict the `output` field, followed by an end-of-sentence (EOS) token. We apply standard causal language modeling masking, where loss is not computed for prompt tokens (labels set to -100). For qualitative evaluation, we use the model's chat template to generate only the assistant's response.

3.2 Models and Quantization

- **Teacher Model**: Qwen/Qwen2.5-3B-Instruct. Used in inference mode with 4-bit NormalFloat (NF4) quantization via the `bitsandbytes` library (Dettmers et al., 2022).
- **Student Model**: Qwen/Qwen2.5-0.5B-Instruct. Trained using QLoRA (Dettmers et al., 2023) with 4-bit base model quantization. LoRA adapters (Hu et al., 2021) are configured with rank $r = 16$, scaling factor $\alpha = 32$, and a dropout rate of 0.05.

To conserve memory, gradient checkpointing is enabled (Chen et al., 2016), and the key-value cache (`use_cache`) is disabled during training.

3.3 Objective Function: Token-Averaged Knowledge Distillation

Our loss function combines the standard cross-entropy (CE) loss on ground-truth labels with a KL divergence loss that encourages the student to match the teacher's probability distribution. The total loss $\mathcal{L}$ is a weighted sum:

$$\mathcal{L} = \lambda_{\text{CE}} \cdot \mathcal{L}_{\text{CE}} + \lambda_{\text{KD}} \cdot \mathcal{L}_{\text{KD}}$$

By default, we set the weights $\lambda_{\text{CE}} = \lambda_{\text{KD}} = 0.5$.

The **cross-entropy loss**, $\mathcal{L}_{\text{CE}}$, is the standard objective for language model training, calculated between the student's predicted logits $\mathbf{z}_S$ and the one-hot ground-truth labels $\mathbf{y}$:

$$\mathcal{L}_{\text{CE}} = \text{CrossEntropy}(\mathbf{y}, \mathbf{z}_S)$$

The **knowledge distillation loss**, $\mathcal{L}_{\text{KD}}$, uses temperature scaling ($T > 1$) to soften the probability distributions from both teacher and student logits ($\mathbf{z}_T$, $\mathbf{z}_S$). This prevents the student from focusing only on the single most likely token and encourages it to learn the richer relational information present in the teacher's full distribution. Following Hinton et al. (2015), the loss is defined as:

$$\mathcal{L}_{\text{KD}} = T^2 \cdot \text{KL}\left(\sigma(\mathbf{z}_T/T) \parallel \sigma(\mathbf{z}_S/T)\right)$$

where σ is the softmax function. The T^2 factor ensures that the gradient magnitudes from the distillation loss remain comparable to the original gradients as temperature changes. We use $T = 2.0$.

A crucial implementation detail is the **reduction method**. Our revised pipeline computes the KL divergence for each token and then **averages the loss over all valid (non-masked) tokens** in a batch.

This contrasts with a sum-based reduction, which can lead to unstable gradients and makes the loss scale dependent on sequence length and batch composition. Token-averaging provides a more stable and interpretable loss signal.

3.4 Optimization

We use the memory-efficient `paged_adamw_8bit` optimizer (Dettmers et al., 2023) with a learning rate of 2×10^{-4}. We train for three epochs on the 5% data split. To accommodate memory limitations, we use gradient accumulation and a maximum sequence length of 512. For stability, we apply **gradient clipping** with a maximum norm of 0.5, which prevents exploding gradients that can occur during training (Pascanu et al., 2013).

4 Experiment 1

4.1 Experimental Setups

We compare two primary configurations to isolate the effect of our methodological improvements:

Baseline (Sum-KL) Our initial setup using a sum-based reduction for the KL divergence loss, without gradient clipping.

Proposed (Avg-KL) Our revised pipeline, which employs **token-averaged KL loss** and adds **gradient clipping** with a max norm of 0.5. All other parameters (models, data, optimizer) are held constant.

4.2 Training Dynamics

Both models converged, but the proposed setup with token-averaged loss exhibited more stable training dynamics. As shown in Figure 1, the loss curve is smoother. Figure 2 separately tracks the cross-entropy and KL divergence components of the loss, which is made possible by the well-scaled, token-averaged objective. This fine-grained logging is critical for diagnosing training and comparing different runs.

4.3 Quantitative Evaluation

We evaluated the student model before and after distillation, along with the teacher model, on the held-out validation set. Table 1 summarizes the results.

The student model trained with our proposed pipeline achieves a final perplexity of **4.085**. This represents a **25.8% reduction** compared to its pre-distillation baseline (PPL 5.504). Remarkably, the 0.5B student not only surpasses the performance of the student from the baseline (Sum-KL) run but also achieves a **3.3% lower perplexity** than the 3B teacher model on this specific data slice.

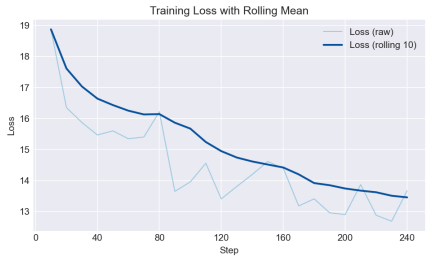


Figure 1: Total training loss versus steps, smoothed with a 10-step rolling average. The proposed (Avg-KL) run shows a stable descent.

Figure 2: Component-wise loss traces for the proposed run, showing the token-averaged CE and KL losses over training steps.

Table 1: Validation metrics on the held-out data slice (`train[5%:6%]`). The proposed Avg-KL run yields the best student performance.

Configuration	Model	Loss (↓)	Perplexity (↓)
Baseline (Sum-KL)	Student (pre-KD)	1.7055	5.5044
	Student (post-KD)	1.5371	4.6513
	Teacher (3B)	1.4410	4.2250
Proposed (Avg-KL)	Student (pre-KD)	1.7055	5.5044
	Student (post-KD)	**1.4070**	**4.0850**
	Teacher (3B)	1.4410	4.2250

4.4 Qualitative Analysis

A qualitative comparison of model outputs (Appendix A) reveals that the distilled student from the proposed run generates more concise and well-structured responses than its pre-distillation counterpart, often matching or improving upon the quality of the teacher's responses for the selected prompts.

5 Experiment 2: Grid Search on Distillation Hyperparameters

Goal. We perform a small but informative grid search to identify a stable recipe under tight memory: we vary the distillation temperature $T \in \{1.5, 2.0\}$, the CE weight $\lambda_{\text{CE}} \in \{0.3, 0.5\}$, learning rate LR $\in \{1\text{e}{-4}, 2\text{e}{-4}\}$, and the maximum sequence length $\{256, 512\}$. LoRA rank is fixed at $r = 16$. The student is `Qwen2.5-0.5B-Instruct` with QLoRA (4-bit); the teacher is `Qwen2.5-3B-Instruct` (4-bit inference). We train for 3 epochs on `yahma/alpaca-cleaned` `train[:5%]` and validate on `train[5%:6%]`.

5.1 Grid and Naming

To maximize readability, each run is identified by an ID **A1–A6**. Detailed hyperparameters are provided in Table 2.

Table 2: Experiment 2 grid (A1–A6) and their hyperparameters.

ID	T	λ_{CE}	LR	Seq	LoRA r
A1	1.5	0.3	2×10^{-4}	256	16
A2	2.0	0.3	2×10^{-4}	512	16
A3	1.5	0.5	2×10^{-4}	256	16
A4	2.0	0.5	2×10^{-4}	512	16
A5	2.0	0.5	1×10^{-4}	512	16
A6	1.5	0.3	1×10^{-4}	512	16

5.2 Evaluation Metrics and Leaderboard

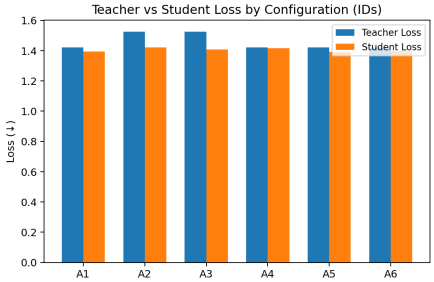
We report validation loss (NLL) and perplexity (PPL) for teacher, student before KD (pre), and student after KD (post). Table 3 summarizes the six runs; Figure 3 visualizes teacher vs. post-student.

Table 3: Experiment 2 results on the held-out split. Lower is better.

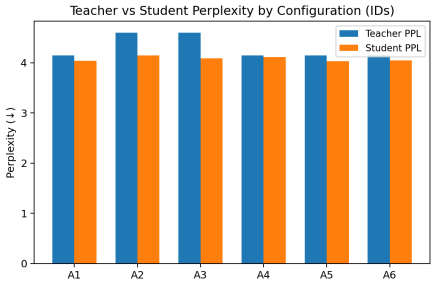
ID	Student (pre)		Student (post)		Teacher	
	Loss	PPL	Loss	PPL	Loss	PPL
A1	1.7807	5.9338	1.4215	4.1433	1.5263	4.6009
A2	1.6940	5.4411	1.4151	4.1168	1.4215	4.1432
A3	1.7807	5.9338	1.4081	4.0883	1.5263	4.6009
A4	1.6940	5.4411	1.3972	4.0438	1.4215	4.1432
A5	1.6939	5.4408	**1.3930**	**4.0270**	1.4215	4.1432
A6	1.6939	5.4408	1.3952	4.0359	1.4215	4.1432

5.2.1 Analysis of Hyperparameter Effects

Our grid search reveals a clear hierarchy of influence. Configuration **A5** achieved the best performance (PPL 4.0270), and the analysis below details the effects of each key hyperparameter based on the results in Table 3.



(a) Teacher vs. Student loss.



(b) Teacher vs. Student perplexity.

Figure 3: Leaderboard-style comparison across configurations (A1-A6).

Sequence Length. The maximum sequence length was the most impactful factor. Configurations with a sequence length of 512 (A2, A4, A5, A6) consistently outperformed those with 256 (A1, A3), as shown in Figure **??**b. This suggests that a larger context window is crucial for the student to learn nuanced distributions from the teacher.

Learning Rate. A lower learning rate of 1×10^{-4} demonstrated more stable and superior performance compared to 2×10^{-4}. Runs A5 and A6, both using the lower LR, yielded the best two perplexity scores. This indicates that a more cautious update step is beneficial in this quantized distillation setup.

Temperature and CE Weight. The effects of distillation temperature (T) and the cross-entropy weight (λ_{CE}) were more subtle. While the optimal run used $T = 2.0$ and $\lambda_{\text{CE}} = 0.5$, the performance differences across their respective settings were marginal. This suggests the pipeline is robust to small variations in these two parameters.

Optimal Recipe. In summary, the optimal and most robust recipe identified by our search is: a **long sequence length (512)**, a **low learning rate**

(1×10^{-4}), a **moderate temperature** ($T = 2.0$), and a **balanced loss weight** ($\lambda_{\mathbf{CE}} = 0.5$).

6 Discussion

The Importance of Loss Normalization and Stability. Our results underscore the critical role of proper loss formulation in knowledge distillation. By averaging the KL divergence loss per token, we decouple the loss magnitude from the number of valid tokens in a batch. This normalization makes the optimization process more stable. Combined with gradient clipping, this approach creates a robust training regime.

On Surpassing the Teacher. The observation that a student model can outperform its teacher is a known phenomenon in KD (Yuan et al., 2020), sometimes referred to as "self-purification" or noise reduction. The distillation process can act as a regularizer, forcing the student to learn a more compressed and potentially more generalized representation. However, it is crucial to note that this outperformance was observed on a small, in-distribution validation set.

7 Future Work and Ablations

This work establishes a solid baseline. We propose the following directions for future investigation:

- **Data Scaling**: Systematically increase the training data from 5% to 100% of the Alpaca dataset.
- **Hyperparameter Sweep**: Conduct a detailed ablation study on the KD temperature T and the cross-entropy weight λ_{CE}.
- **Architectural Choices**: Evaluate the trade-offs between LoRA rank and maximum sequence length.
- **Intermediate Representation Distillation**: Augment the logit-based KD with an auxiliary loss that matches intermediate hidden states, inspired by methods like TinyBERT (Jiao et al., 2020).

8 Conclusion

We have presented a stable and resource-efficient pipeline for knowledge distillation, successfully compressing a 4-bit 3B teacher model into a 4-bit 0.5B student model. Our key finding is that employing a **token-averaged KL divergence loss** with gradient clipping is markedly superior to a sum-based approach, leading to a significant **25.8% reduction in perplexity**. The resulting 0.5B model demonstrated performance that slightly exceeded its 3B teacher on an in-distribution validation set. This work provides a practical and reproducible recipe for creating powerful, lightweight language models.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. 2016. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, pages 74338–74365.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. 2022. A survey of quantization methods for efficient neural network inference. *Low-Power Computer Vision*, pages 291–335.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174.

Yann LeCun, John S Denker, and Sara A Solla. 1990. Optimal brain damage. In *Advances in neural information processing systems*, volume 2.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*.

Li Yuan, Francis E H Tay, Guilin Li, Tao Wang, and Jiashi Feng. 2020. Revisiting knowledge distillation: A teacher-free approach. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3379–3388.

Appendix

A Reproducibility Checklist

- **Hardware**: Single NVIDIA P100 GPU for 2-3 Hours (T4/3060/4060 class or equivalent).
- **Quantization**: Teacher with 4-bit inference; Student with 4-bit QLoRA.
- **Data**: `yahma/alpaca-cleaned`, with specified train/validation splits.
- **Optimizer**: `paged_adamw_8bit` with LR $\approx 2 \times 10^{-4}$.
- **Distillation Parameters**: Temperature $T \approx 2.0$; Loss weights $\lambda_{\text{CE}}, \lambda_{\text{KD}} \approx 0.5$.
- **Key Flags**: Gradient checkpointing enabled; `use_cache=false` during training.
- **Loss Reduction**: KL divergence must be **averaged per token**.
- **Stability**: Gradient clipping with max norm ≈ 0.5.

B Use of AI-Based Tools

This appendix documents the use of artificial intelligence (AI)-based tools in the preparation of this academic work.

List of Steps Involving AI-Based Tools

- **DeepSeek**: I consulted DeepSeek models to learn more formal organization of an conclusion chapter. The suggested frameworks were adapted and rewritten entirely in my own words.
- **QuillBot**: QuillBot was used sparingly to rephrase sentences for improved readability and flow. All suggestions were manually reviewed and edited to ensure alignment with my original intent and academic style.
- **DeepL and Youdao Translation**: DeepL and Youdao Translation assisted in translating a small number of technical terms and short phrases from Chinese to English to clarify meaning during drafting. These translations were verified and incorporated into my own text.